1. 向我们总结此项目的目标以及机器学习对于实现此目标有何帮助。作为答案的部分，提供一些数据集背景信息以及这些信息如何用于回答项目问题。你在获得数据时它们是否包含任何异常值，你是如何进行处理的？【相关标准项：“数据探索”，“异常值调查”】

答：项目目标是通过安然数据集发现嫌疑人，机器学习可以通过学习数据的特征锁定嫌疑人。数据点总数为 146，每个人有 20 个特征，目前一共标注了 18 个嫌疑人，数据集中工资、邮箱、薪酬总额、董事费等特征有较多的缺失值，目标值就是是否是嫌疑人。我在分析财务数据 salary 和 bonus 之间的关系时发现了一个异常值，奖金和薪水远远高于其他人，审查发现一共有三个异常值。第一个为 TOTAL，很明显不是一个人名，将他作为真正的异常值用 pop 方法删除掉，剩下两个分别是 SKILLING JEFFREY K 和 LAY KENNETH L，分别是安然公司的 CEO 和董事长，他们不是真的异常值，因此不对他们进行处理。

2. 你最终在你的 POI 标识符中使用了什么特征，你使用了什么筛选过程来挑选它们？你是否需要进行任何缩放？为什么？作为任务的一部分，你应该尝试设计自己的特征，而非使用数据集中现成的——解释你尝试创建的特征及其基本原理。（你不一定要在最后的分析中使用它，而只设计并测试它）。在你的特征选择步骤，如果你使用了算法（如决策树），请也给出所使用特征的特征重要性；如果你使用了自动特征选择函数（如 SelectBest），请报告特征得分及你所选的参数值的原因。【相关标准项：“创建新特征”、“适当缩放特征”、“智能选择功能”】

答：**特征缩放**：先对数据使用 preprocessing.MinMaxScaler 缩放，因为当有多个特征向量的时候，如果其中一个变化范围比较大，该特征向量的参数可能会变化范围很大，会不合理地加大权重，如果将所有特征向量的变化范围维持在一个标准化范围之中，就能减小该特征向量的影响程度。SVM，K-means 算法尤其需要特征缩放。SVM 算法要找一个分割面把点分开，计算距离时，是用一个维度和另一个维度间做交换。如果某一点是其他点的两倍，距离值也会变成两倍。K-means 算法是计算每个点到集群中心的距离，如果一个变量扩大一倍，它的数值也会扩大一倍。而决策树和线性回归不受特征缩放的影响。决策树的分割是一系列的水平线和垂直线，不存在两者的交换。在考虑一个维度时，不需要考虑另一个维度的值。如果把一个维度进行缩放，分割的位置会变化，但顺序不会变。线性回归的每个特征有一个相应的系数，系数和特征总是一起出现。一个特征的变化不会影响到另一个特征的系数。

特征选择: 我在 poi_feature_select.py 里实现特征的选择。使用 SelectKBest 选择了 5 个重要特征，分别是'exercised_stock_options', 'total_stock_value', 'salary', 'deferred_income', 'expenses'。选择这 5 个特征是因为它们得分较高，缺失值较少，选择不同个数的特征的算法性能如下图所示，可以看出选择这 5 个特征的三个评价指标综合来看是最好的。

使用特征	accuracy	recall	precision
'exercised_stock_options', 'total_stock_value'	0.82008	0.23550	0.36768
Bonus.total_stock_value exercised_stock_options	0.80477	0.39000	0.37178
'exercised_stock_options', 'total_stock_value', 'salary', 'deferred_income'	0.84843	0.19850	0.43341
'exercised_stock_options', 'total_stock_value', 'salary', 'deferred_income', 'expenses'	0.84280	0.35900	0.40022
'exercised_stock_options', 'total_stock_value', 'bonus', 'salary', 'deferred_income', 'expenses'	0.82287	0.33850	0.33665
'exercised_stock_options', 'total_stock_value', 'bonus', 'salary','deferred_income','expenses','long_term_incentive'	0.83467	0.28250	0.35093

新特征创建：创建了每个人收到嫌疑人的邮件占总收件之比和每个人发给嫌疑人的邮件占总发件之比这两个新特征。基本原理就是定义函数计算带有 poi 标识的邮件占总邮件的比率。通过可视化作图可以看出，两个新特征比率高于一定值就有很大可能是嫌疑人，这方便了机器学习算法。但通过测试算法也可以发现此新特征决策树算法的准确性影响不大，甚至有所下降（如下图所示）。

使用特征	accuracy	recall	precision
未使用新特征	0.84280	0.35900	0.40022
加上新特征"to_poi_ratio","from_poi_ratio"	0.84293	0.32150	0.39160

3. 你最终使用了什么算法？你还尝试了其他什么算法？不同算法之间的模型性能有何差异？【相关标准项：“选择算法”】

我最终选择了 DecisionTreeClassifier 算法。在此之前我还尝试了 GaussianNB。经 tester.py 测试，GaussianNB 算法的性能为 Precision: 0.45103、Recall: 0.22951，决策树为 Precision: 0.35806、Recall: 0.40197，综合来看决策树更好，猜测经过调整参数后决策树的性能会更优。贝叶斯算法容易忽略属性之间的关联，更适合文本分析，而决策树的分类效率更好，不过容易过拟合，所以之后要进行交叉验证。

4. 调整算法的参数是什么意思，如果你不这样做会发生什么？你是如何调整特定算法的参数的？（一些算法没有需要调整的参数 – 如果你选择的算法是这种情况，指明并简要解释对于你最终未选择的模型或需要参数调整的不同模型，例如决策树分类器，你会怎么做）。【相关标准项：“调整算法”】

   机器学习的模型都是参数化的，以便于其针对特定的问题进行调整。一个模型有很多参数，寻找这些参数的最佳组合其实是一个搜索问题。如果不调整参数可能会造成过拟合等问题，无法最好的训练和预测。我最终选择了决策树算法，使用 gridsearchcv 对参数进行调整，更改了一些默认参数，gridsearchcv 使用三折交叉验证筛选出最优参数，评分方法为平衡 F 分数。

5. 什么是验证，未正确执行情况下的典型错误是什么？你是如何验证你的分析的？【相关标准项：“验证策略”】

验证是用于评估模型好坏的一个重要方法，通常将数据集分为训练集和测试集就是为了验证。前者用以建立模型，后者则用来评估该模型对未知样本进行预测时的泛化能力。未正确执行情况下的典型错误是过拟合，导致分类器在训练集上的正确率可能高至 100%，但在测试集上的得分却极差。我使用 tester.py 里的验证方法，即 StratifiedShuffleSplit 交叉验证，它的工作原理就是先将样本随机打乱，然后根据设置参数划分出 n 组 train/test 对，每组里的 train 和 test 比例都相同，用这些 train/test 对进行分类预测，然后经过很多次的迭代，把所有的预测分数求平均，这样能避免随机性的影响，防止过拟合。

6. 给出至少 2 个评估度量并说明每个的平均性能。解释对用简单的语言表明算法性能的度量的解读。【相关标准项：“评估度量的使用”】

   我使用了精确**度**、召回**率**这两个评估度量，精确度计算的是正确被预测为 poi 的数量与所有被标记为 poi 的数量之比，召回率计算的是正确被预测为 poi 的数量与真实的全部 poi 数量之比。最终选择的优化后决策树算法的精确度为 0.40013，召回率为 0.31650。